Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 93 (2016) 41 – 47

Procedia Computer Science

6th International Conference On Advances In Computing & Communications, ICACC 2016, 6-8 September 2016, Cochin, India

Reliability Prediction of Web Services using HMM and ANN models

Suhas Honamore[a,*], Kapil Dev[a], Ranjana Honmore[b]

[a] *Dept. of CS&E, National Institute of Technology, Rourkela-769008, India*
[b] *J E Society K A Lokapur College, Athani-591304, India*

Abstract

Service-oriented architecture (SOA) is an approach for constructing distributed business-driven frameworks. Reliability of service-oriented systems (SOS) relies on the Web services used and in addition Internet associations. Predicting reliability of Web service has turned into an imperative exploration issue. In this paper Hidden Markov Model (HMM) and Artificial Neural Network (ANN) are used to model the Web service failure model and predict Web services reliability. The forward-backward estimation-maximization algorithm is used to estimate the modeling parameters for HMM. Different methods of ANN like feed-forward, multilayer perceptron, and radial basis function neural networks are applied to predict the reliability. The accuracy for each model is calculated and the performance parameters of models are compared by considering Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) metrics.

Peer-review under responsibility of the Organizing Committee of ICACC 2016

Keywords: Artificial Neural Network; Hidden Markov Model; Web Service; Reliability

1. Introduction

Web based applications are keep running without introducing to local frameworks or systems. This concept helps to innovate the development of various Web services in order to meet the requirement of different business frameworks. For service-oriented architecture (SOA)[1] arrangement, Web service gives a superior stage to exchange information and to invoke outside applications. By discovering required Web services and by integrating them, service-oriented system (SOS) can be efficiently built.

SOS has been largely pursued in different domains, such as automotive systems, business-to-business collaboration, multi-media services, etc. Reliability is an important aspect in quality of service (QoS) and the reliability of these systems mainly relies on the Web services invoked in the systems. To select reliable Web service and to integrate in SOS, the prediction of reliability of Web service guides in subsequent activities. The ability of predicting reliability at the early phase helps to cut-down the cost of re-engineering and helps to produce more reliable SOS.

* Corresponding author. Tel.: +91-903-577-8340.
E-mail address: suhas103@gmail.com

Peer-review under responsibility of the Organizing Committee of ICACC 2016
doi:10.1016/j.procs.2016.07.179

Presently, a number of stochastic models are available to identify the software failure rates. These models depend on refined presumptions and the objective of these models is to judge the software reliability by considering the perception on past failures and the corrections made. The study shows the failure process of these kind of software are equivalently any of these random processes $\{N_t\}_{t>0}$, $\{X_i\}_{i>0}$, or $\{T_i\}_{i>0}$; where N_t indicate the failures arrived between time 0 and t; T_i, $i > 0$ is the time of successive software failures, with $T_0 = 0$ and $X_i = T_i - T_{i-1}$, $i > 0$, denote the time between consecutive software failures. Most of the software reliability models assume that the number of failures arrived between time 0 and t, $\{N_t\}_{t>0}$ is a non-homogeneous Poisson process (NHPP), for which failure rates are deterministic functions of time: $\lambda_t = \lambda(t)$. However, NHPP models expect a rectification for every failure, and the effectiveness of debugging is homogeneous in time.

In the circumstances of SOA, the reliability of SOS framework depends on the framework, and mainly relies on the Web services invoked. The distinctive service clients might encounter very distinctive reliability for same Web service, which makes a task difficult to build a reliability model for SOS. The assessment of reliability of Web service can be studied using Hidden Markov Model (HMM), because HMM focuses on discontinuous nature of failure rate. Further, another model, being widely used for reliability analysis is the Artificial Neural Network (ANN) model.

ANN is a versatile system that takes input, output data and analyze the relationships between them. By these relations, it can predict unobserved data having same characteristics as that of input data. Different types of ANN exists; among them feed-forward, multilayer perceptron (MLP) and radial basis function (RBF) are applied in this study. Here Web service's throughput and response time are taken as input and number of successful access are taken as output for neural network.

Rest of this paper is organized as follows. Related work for this study is discussed in Section 2. Section 3 deals with the HMM for software reliability process, model parameter estimation, hidden states' restoration, and model selection based on the Bayesian Information Criterion (BIC) values. Section 4, gives the brief explanation about ANN methods applied. Section 5 contains the experiments performed on Web service data and results are compared with reliability models. Finally Section 6 concludes the study.

2. Related Work

To predict reliability of Web service there exist many different methods. Yingxin Ren et al.[2] proposed a discrete time Markov chain (DTMC) based approach to predict reliability of Web service composition. Bixin Li et al.[3] proposed an approach to predict reliability of Web service based on static BPEL code structure analysis and dynamic run-time information. But BPEL's code structures are generally not available to service consumers. Marin Silic et al.[4] presented a model called CLUStering for reliability prediction of atomic Web services which reduce the execution time for prediction. Ning Huang et al.[5] presented algebra-based reliability prediction by considering architecture and operational profile in design stage of Web service compositions.

Chen et al.[6] proposed an approach where survivability of SOA is treated as a multidimensional QoS property, and it is evaluated using HMM. G. Rahnavard et al.[7] used HMM to detect the Web services anomaly detection, which improves quality of Web services security and attack detection. V. Grassi et al.[8] presented a methodology, which identifies the reliability based on the information published by each assembled service. All service providers don't publish information related to reliability; so it is not a easy task to identify the reliability. Malik et al.[9] proposed a method to predict the reputation of Web service using HMM, where rater feedbacks are not available.

Eyhab Al-Masri[10] proposed a framework to discover efficient Web service using ANN. Zhang Jin-hong[11] applied RBF method to predict the QoS of Web service. Lianzhang Zhu and Xiaoqing Liu[12] introduced a technique using ANN for SOS where quality function deployment is used with relationships between design attributes and quality of service requirements.

3. Hidden Markov Model as Reliability model

3.1. Modeling of failure process with HMM

To use HMM framework, it's essential to assume the rate of failure process Λ, is of finite set with N elements in it. Process which meets following assumptions is a hidden Markov chain.

- Λ is a discrete-valued Markov chain;
- times between failures $\{X_N\}_{N\geq 1}$ are independent; like for a Web service failures may rise in different time to different users
- X_N has an exponential distribution with parameter $\lambda^{(j)}$; as Web service failure caused from server side rises the failure rate to all users

3.2. Parameter estimation

Standard procedures for evaluating HMM parameters include batch learning, implemented with specific Expectation - Maximization (EM) technique or the Baum-Welch (BW) algorithm. In both cases, parameters of HMM are evaluated by more than a few training iterations, till the objective function (e.g., maximum likelihood) is maximized.

In HMM complete data $y_1^n = (x_1^n, \lambda_1^n)$ is divided into observed data x_1^n and missing data λ_1^n. This missing data complicates the parameter estimation η by likelihood maximization. EM algorithm is used to restore the missing data, which works in following steps:

- Estimation (E) step: determines the function Q

$$Q(\eta, \eta^{(m)}) = E_{\eta^{(m)}}[logP_\eta(\Lambda_1^n, X_1^n = x_1^n) \mid X_1^n = x_1^n] \tag{1}$$

- Maximization (M) step: maximizes the function Q in respect of η

$$\eta^{(m+1)} = arg \max_{\eta} Q(\eta, \eta^{(m)}) \tag{2}$$

The sequence of maximization tends towards to consistent solution after running it for some iteration; in this paper 50 iterations are performed to get consistent solution. The distribution values can be calculated by running the BW algorithm, first from start time to forward and second from end time to backward; so this is called as *forward-backward* algorithm.

3.3. Hidden states Restoration

One can restore the unknown states in the context of HMM, by giving values to an unknown state sequence λ_1^n, known as hidden states. A characteristic approach to restore the hidden states is that process for most likely values by considering known sequence values x_1^n. This results in maximum a posteriori, which can be done by Viterbi algorithm, computed as:

$$arg \max_{\lambda_i^\eta} P_\eta(\Lambda_1^n = \lambda_1^n \mid X_1^n = x_1^n) \tag{3}$$

3.4. Bayesian Information Criterion

Models having moderate complexity are difficult to judge, to overcome this criterion based on log-likelihood parameter: Bayesian Information Criterion (BIC) [13] is used for selecting a best model among the set of given models. The lowest BIC value for a model is preferred as best model. The criterion is given as follows:

$$BIC(K) = \log(P_{\eta\hat{K}}(x_1^n)) - \tfrac{v_K}{2}\log(n) \tag{4}$$

where $\log(P_{\eta\hat{K}}(x_1^n))$ represents a maximum log-likelihood with K hidden states; n represents length of observed failure sequence; and v_K represents number of independent parameters in η_K

4. Artificial Neural Network (ANN)

Modeling of artificial neurons is done by abstracting complexity of neurons. These models comprise of inputs, which are duplicated by weights (intensity of the each signal), then activation of neurons are computed using mathe-

matical functions. The outputs of neurons are calculated (it may depends on some threshold value) by using different functions.

The computation of the neuron differs as weights of neurons differ. By altering the different weights of neuron, one can get the required output. Different algorithms are used to adjust the weights in order to get desired outputs. The neuron weight adjusting process is known as leaning or training.

4.1. Feed-forward neural network (FNN)

The feed-forward ANN uses back-propagation algorithm. In this algorithm the neurons are composed in layers, and send their signs forward, and errors disseminated backward. The back-propagation calculation utilizes supervised learning, which implies that known inputs with corresponding outputs are give to compute network.

The back-propagation algorithm uses the weighted sum as activation function (F) i.e. sum of the inputs x_i multiplied by their corresponding weights wt_{ji}:

$$F_j(\overline{x}, \overline{wt}) = \sum_{i=0}^{n} x_i wt_{ji} \tag{5}$$

In FNN, sigmoidal function is most commonly used as output function, which is given as follows:

$$O_j(\overline{x}, \overline{wt}) = \frac{1}{1+e^{F_j(\overline{x},\overline{wt})}} \tag{6}$$

The objective of training process is to get a required output after giving certain inputs. To reduce the error (difference between the actual and desired output) weights are supposed to adjust. The error of overall network is calculated by adding all the error of neurons, which is given as below:

$$E(\overline{x}, \overline{wt}, d) = \sum_j (O_j(\overline{x}, \overline{wt}) - d_j)^2 \tag{7}$$

To reduce error, back-propagation algorithm calculates how errors are related to input values, their weights, and to output values. The weights are adjusted using gradient descendent method, with training rate of η which is given as:

$$\Delta wt_{ji} = -\eta \frac{\partial E}{\partial wt_{ji}} \tag{8}$$

4.2. Multilayer Perceptron (MLP) neural network

A multilayer perceptrons (MLP) neural network comprises of different layers of neurons with each layer completely connected to the next layer. MLP expands the perceptrons function with hidden layers i.e., layers of processing elements which are not directly associated with outputs. In MLP except inputs, all neurons are considered by non-linear activation function. MLP is a modified version of linear perceptron, which can differentiate data those are not linearly separable. Two most commonly used non-linear functions are hyperbolic tangent (*tanh*) and *sigmoid* functions. This study considers sigmoid function as activation function.

4.3. Radial basis function (RBF) neural network

RBF neural network is a powerful method in multidimensional space. A RBF is used as substitute for sigmoidal function of hidden layer in multi-layer perceptrons. For RBF, different functions are tested as activation functions, but Gaussian function is most suited one and it is given as follows:

$$\Phi_j(X) = exp\left[-(X-\mu_j)^T \sum_j^{-1} (X-\mu_j)\right] \tag{9}$$

where X indicates input features vector, $\sum_j$ indicates the covariance matrix of j^{th} Gaussian function, and μ_j is mean of the j^{th} Gaussian function.

The RBF networks have the benefit of not experiencing minima as MLP suffers from it; because in RBF learning process, parameters are adjusted from hidden layer to output layer in linear mapping. Linearity make sure the error surface as quadratic; so the minimum is effectively discovered.

5. Experiments

5.1. Datasets

To predict reliability, the dataset is considered from Quality of Service (QoS) dataset[14]. In addition Web services having similar functionality are identified and invoked at-least 100 times from different computers. The overall dataset has 100 Web services invoked by 100 different users at-least 150 times. Depending on the invocation, the failure rate for each Web service is calculated and they are arranged in the form of matrix of size 100x100. This matrix has been used as a dataset for this study.

For HMM model, dataset is divided into 20 different groups (DG1 to DG20), each group contain 5 similar functionality Web services. There is a possibility that user can switch to other same functionality Web service if the invoked Web service is failed. Here, this transition probability of Web services is denoted by 5x5 matrix, known as transition probability matrix.

To evaluate ANN, throughput and response time are used as input to ANN methods. The successful invocation of Web service is considered as output in supervised learning methods.

The different prediction methods HMM and ANN are evaluated by randomly removing some percentage of data from dataset. The unremoved data is referred as data density (DD) for prediction, while removed data is used for validation purpose.

5.2. Metrics

The performance of different methods can be compared using different metrics. In this study, Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) metrics are used along with the percentage of accuracy to compare the implemented models.

The difference between predicted value and the corresponding observed value is calculated for testing data set. The average of these value gives the MAE of data set. The mathematical equation for MAE is given as follows:

$$MAE = \frac{\sum_{u,i} |p_{u,i} - \hat{p_{u,i}}|}{N} \tag{10}$$

where N denotes the number of predicted values, $\hat{p_{u,i}}$ indicates predicted failure probability for Web service, and $p_{u,i}$ indicates the corresponding observed value by user u for i^{th} Web service.

For every test data, square of difference between predicted value and the corresponding observed value is calculated and then averaged for all data set. The square root of the averaged data gives the RMSE for data set, which is given as below:

$$RMSE = \sqrt{\frac{\sum_{u,i} (p_{u,i} - \hat{p_{u,i}})^2}{N}} \tag{11}$$

5.3. Hidden Markov Model

The HMM is applied on dataset, while the description for HMM implementation on first data group (DG1) is given here. In HMM, for a particular problem, identifying number of hidden states is not an easy task. So the first step is to find the number of hidden states K for data group DG1. To find that the HMM is executed with K_{min} = 1 to K_{max} = 7 and BIC values are calculated for them. It has been observed that BIC value is less with K = 5, which fits the complexity of data. Same procedure is followed for all data groups, the result shows that maximum data groups (19 out of 20) have less BIC values with K = 5. Hence, throughout experiment the number of hidden states used is 5. The transition probability matrix used for dataset group DG1 is shown in Table-1. Depending on transition matrix, the probability matrix converges to particular valued matrix after n^{th} iteration. So initially, the probability matrix is considered with equal probability for all Web services.

The emission matrix (EMS_MAT) for a group is obtained based on the failure to invoke for the Web services by using 5x100 matrix. For WS1, 15128 number of invocation are made by 100 different users. And for user1, 11 invocations failed, which leads to an entry of 0.00072712 in emission matrix for DG1; i.e. EMS_MAT (1, 1) = 0.00072712.

Table 1. Transition Matrix for DG1

	WS1	WS2	WS3	WS4	WS5
WS1	0.700	0.100	0.100	0.050	0.050
WS2	0.070	0.720	0.055	0.055	0.100
WS3	0.050	0.050	0.800	0.070	0.030
WS4	0.045	0.045	0.040	0.750	0.120
WS5	0.040	0.030	0.030	0.050	0.850

Based on transition and emission matrices, sequence of states and sequence of emission symbols are generated. Running EM algorithm on these matrices in an iteration of 50, most likelihood states are identified. Using Viterbi algorithm, most likelihood states are restored.

Using restored states along with known states the logarithmic likelihood probability failure is calculated. Then the MAE and RMSE values for data group DG1 is calculated by using equation 10 and 11 respectively. Same procedure is followed for remaining groups and MAE, RMSE values for them are noted.

Following are the result of this study:

- For all data groups, selecting number of hidden states are extremely vital. This study has identified number of hidden states as 5 i.e. K=5 for a group. However this number is quite large, which observes the failure rate as very closely to the transitions
- For all data groups, the restoration of hidden states detected are homogeneous; which leads to identification of major corrections in the Web service provider

5.4. ANN models

In this study, throughput and response time of 100 Web services for all 100 users are used as input data for neural networks and output layer uses the successful invocation of Web services. First, the data is normalized, then it is divided into testing and training data sets. For ANN data set, the data density indicates data set taken for training. For example, Data density DD = 90 % means 90% of data are used for training and 10% for testing.

The feed-forward neural network with a back-propagation algorithm is implemented to predict the reliability of Web services. The number of hidden neurons are identified by a trial and error mechanism that is running FNN with different number of hidden neurons. It has been found that FNN gives better result with 10 hidden neurons. The supervised training mechanism is used with learning rate of 0.12 .

An implementation for Multilayer Perceptron (MLP) is done by fully connected Neural Network with a sigmoid unipolar activation function. The number of hidden neurons identified for MLP are 10. The training is done by using the Back-propagation algorithm. The training aspect stops when maximum number of epochs (100) is reached and the learning rate used is 0.15.

In RBF, neural network is trained by using three main steps: (i) prototype is selected through k-means clustering by running it by at most 100 iterations; (ii) the width of each RBF neurons i.e., beta coefficient is calculated based on the average distance between a cluster's data points and its center. The average distance is known as sigma (σ), and beta coefficient (β) is given by $\beta = \frac{1}{2\sigma^2}$; (iii) using gradient descent, output weights are trained. The RBFN's accuracy is evaluated on the training set by considering original dataset with the placements of prototype data.

To compare all methods of ANN with HMM models, MAE and RMSE are calculated for them along with prediction accuracy. The comparison of these metrics for HMM and ANN methods with different data density are shown in Table-2. The accuracy of each model is listed in Table-3

From Table-2, it is observed that the HMM and FNN are almost same with low data density (e.g. 50%, 60% and 70%) and as data density increases (e.g. 80% and 90%) both give the same result. Finally, the RBF method obtains the best prediction accuracy that is 94.7% and smaller RMSE and MAE values for all data sets considered in the study.

Table 2. MAE and RMSE values of all models

Methods	DD=50%		DD=60%		DD=70%		DD=80%		DD=90%	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
HMM	0.2912	0.4534	0.2012	0.3606	0.1697	0.3147	0.1412	0.2862	0.1209	0.2493
FNN	0.2895	0.4457	0.1996	0.3523	0.1685	0.3110	0.1426	0.2853	0.1212	0.2493
MLP	0.2397	0.3948	0.1443	0.3149	0.1114	0.2813	0.0924	0.2461	0.0845	0.2187
RBF	0.1755	0.3451	0.1014	0.2652	0.0703	0.2184	0.0650	0.1927	0.0603	0.1468

Table 3. Accuracy of models

Methods	HMM	FNN	MLP	RBF
Accuracy	90.2%	90.4%	92.6%	94.7%

6. Conclusion

The reliability of SOS mainly depends upon the Web service(s) used in it. However, it is a not an easy task to find a good model which identify the reliability of Web services. In this paper, HMM and different ANN methods like FNN, MLP, and RBF neural network are used to predict the failure rate of Web service. Experimentation is likewise done on real Web service dataset. For Hidden Markov model 20 different groups of similar functionality Web services are used as dataset. Same Web service's, the throughput and response time along with successful invocation Web service is used as data set for ANN methods. It has been observed that radial basis function neural network gives the better accuracy compared to HMM and other ANN methods. In future, the study can be carried out by combining HMM and ANN to get better accuracy.

References

1. D. Sprott, L. Wilkes, Understanding service-oriented architecture, The Architecture Journal 1 (1) (2004) 10–17.
2. Y. Ren, Q. Gu, J. Qi, D. Chen, Reliability prediction of web service composition based on dtmc, in: Secure Software Integration and Reliability Improvement, 2009. SSIRI 2009. Third IEEE International Conference on, IEEE, 2009, pp. 369–375.
3. B. Li, X. Fan, Y. Zhou, Z. Su, Evaluating the reliability of web services based on bpel code structure analysis and run-time information capture, in: Software Engineering Conference (APSEC), 2010 17th Asia Pacific, IEEE, 2010, pp. 206–215.
4. M. Silic, G. Delac, S. Srbljic, Prediction of atomic web services reliability based on k-means clustering, in: Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering, ACM, 2013, pp. 70–80.
5. N. Huang, D. Wang, X. Jia, An algebra-based reliability prediction approach for composite web services, in: Software Reliability Engineering, 2008. ISSRE 2008. 19th International Symposium on, IEEE, 2008, pp. 285–286.
6. L. Chen, Q. Wang, W. Xu, L. Zhang, Evaluating the survivability of soa systems based on hmm, in: Web Services (ICWS), 2010 IEEE International Conference on, IEEE, 2010, pp. 673–675.
7. G. Rahnavard, M. S. Najjar, S. Taherifar, A method to evaluate web services anomaly detection using hidden markov models, in: Computer Applications and Industrial Electronics (ICCAIE), 2010 International Conference on, IEEE, 2010, pp. 261–265.
8. V. Grassi, S. Patella, Reliability prediction for service-oriented computing environments, Internet Computing, IEEE 10 (3) (2006) 43–49.
9. Z. Malik, I. Akbar, A. Bouguettaya, Web services reputation assessment using a hidden markov model, in: Service-Oriented Computing, Springer, 2009, pp. 576–591.
10. E. Al-Masri, Q. H. Mahmoud, Discovering the best web service: A neural network-based solution, in: Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on, IEEE, 2009, pp. 4250–4255.
11. Z. Jin-hong, A short-term prediction for qos of web service based on rbf neural networks including an improved k-means algorithm, in: Computer Application and System Modeling (ICCASM), 2010 International Conference on, Vol. 5, IEEE, 2010, pp. V5–633.
12. L. Zhu, X. F. Liu, Technical target setting in qfd for web service systems using an artificial neural network, Services Computing, IEEE Transactions on 3 (4) (2010) 338–352.
13. I. Chang, S. W. Kim, Modelling for identifying accident-prone spots: Bayesian approach with a poisson mixture model, KSCE Journal of Civil Engineering 16 (3) (2012) 441–449.
14. Z. Zheng, M. R. Lyu, Personalized reliability prediction of web services, ACM Transactions on Software Engineering and Methodology (TOSEM) 22 (2) (2013) 12.